AMENDMENTS TO THE CLAIMS

Please amend the claims as follows:

1.   (Currently Amended) A method performed by a computer system comprising:

providing a test module;

detecting a test module interface associated with the test module, the test module interface being created as an EFI protocol for enabling programs to detect and interact with the test module interface; and

calling a function identified by the test module interface to cause a first test configuration of the test module to be created.

2.   (Original) The method of claim 1, further comprising:

detecting a test routine associated with the test module using the first test configuration; and

causing the test routine to be executed.

3.   (Original) The method of claim 1, further comprising:

loading the test module;

detecting a device associated with the test module; and

in response to detecting the device, creating the test module interface.

4.   (Original) The method of claim 3, further comprising:

detecting a change associated with the device; and

in response to detecting the change, reinstalling the test module interface associated with the test module.

5.   (Original) The method of claim 4, further comprising:

in response to detecting the change,

calling the function to cause a second test configuration of the test module

to be created.

6.   (Original) The method of claim 1, further comprising:

registering a use of the test module by a program.

7.   (Original) The method of claim 6, further comprising:

unloading the test module; and

informing the program of the unloading prior to unloading the test module.

8.   (Original) The method of claim 7, further comprising:

conveying a defer signal from the program to the test module; and

in response to the defer signal, canceling the unloading of the test module.

9.   (Currently Amended) A computer system comprising:

a processor; and

a memory coupled to the processor and including a program and a test

module;

the program being executable by the processor to:

detect a test module interface associated with the test module, the

test module interface being created as an EFI protocol for enabling

programs to detect and interact with the test module interface; and

call a function identified by the test module interface to cause a first

test configuration of the test module to be created.

10. (Original) The computer system of claim 9, wherein the program is executable to:

detect a test routine associated with the test module using the first test configuration; and

cause the test routine to be executed.

11. (Original) The computer system of claim 9, wherein the test module is executable to:

detect a device associated with the test module; and

in response to detecting the device, create the test module interface.

12. (Original) The computer system of claim 11, wherein the test module is executable to:

detect a change associated with the device; and

in response to detecting the change, reinstall the test module interface associated with the test module.

13. (Original) The computer system of claim 12, wherein the program is executable to:

call the function identified by the test module interface to cause a second test configuration of the test module to be created.

14. (Original) The computer system of claim 9, wherein the program is executable to:

cause a use of the test module by the program to be registered.

15. (Original) The computer system of claim 14, wherein the test module is executable to:

cause the test module to be unloaded; and

cause the program to be notified prior to unloading the test module.

16.   (Currently Amended) The computer system of claim 15, wherein the program is executable to:

convey a defer signal from to the test module from the program to the test module; and

wherein the test module is executable to:

in response to the defer signal, cause the unloading of the test module to be canceled.

17.   (Currently Amended) A computer system comprising:

a processor; and

a memory coupled to the processor and including a first program and a second program that includes an interface;

the first program being executable by the processor to:

detect the interface, the interface being created as an EFI protocol for enabling the programs to detect and interact with the interface;

call a first function identified by the interface, the first function being executable by the processor to register a use of the second program by the first program.

18.   (Original) The computer system of claim 17, wherein the first function is executable by the processor to cause an entry associated with the first program to be stored in the memory.

19.   (Original) The computer system of claim 18, wherein the entry includes a first identifier associated with the first program, a second identifier associated with the

7

interface, and a third identifier associated with a second function.

20.   (Original) The computer system of claim 17, wherein the second program is executable by the processor to:

cause the second program to be unloaded from the memory; and

cause the first program to be notified that the second program seeks to be unloaded.

21.   (Original) The computer system of claim 20, wherein the first program is executable by the processor to:

convey a defer signal to the second program in response to being notified that the second program seeks to be unloaded; and

wherein the second program is executable by the processor to:

cause the second program not to be unloaded in response to receiving the defer signal.

22.   (Original) The computer system of claim 17, wherein the first program is executable by the processor to:

call a second function identified by the interface, the second function being executable by the processor to cause a configuration of the second program to be created.